# A membership management protocol for mobile P2P networks

Mohamed Karim Sbai, Emna Salhi, Chadi Barakat

# A membership management protocol for peer-to-peer services in MANET

Mohamed Karim Sbai
EPI Planete, INRIA, France
Email: mksbai@sophia.inria.fr

Emna Salhi
EPI Planete, INRIA, France
ENSI, Tunisia
Email: esalhi@sophia.inria.fr

Chadi Barakat
EPI Planete, INRIA, France
Email: cbarakat@sophia.inria.fr

***Abstract*—MANETs are self-organizing networks composed of mobile wireless nodes with often scarce resources. Distributed applications based on the P2P paradigm are the best candidates to run over such networks. To profit from the service provided by a P2P overlay (e.g. file sharing using BitTorrent), a node needs to be permanently informed about the other members of the overlay (e.g. other peers interested in the same file as currently provided by the BitTorrent central tracker). This P2P membership management is a costly and difficult task in such dynamic and resource limited environment. We focus on this problem and we propose a robust, network friendly and decentralized membership management protocol allowing peer discovery and update. Compared to flooding, client-server or multicast based approaches, our protocol achieves significantly lower network overhead and lower pollution of caches caused by peers who have left. Moreover, as network splits are very frequent in MANETs, our protocol is designed to be partition-aware. Namely, it allows separate overlays providing the same service to efficiently merge together when communication opportunities occur. The efficiency of our solution is validated through extensive NS-2 simulations.**

## I. Introduction

The wide spread of mobile devices (Laptops, PDAs, Smartphones, etc) encourages users to connect directly to each other to form ad hoc communities. These devices, forming spontaneous wireless multi-hop networks thanks to the use of ad hoc routing protocols, can be used to run several services such as content sharing, multimedia streaming, instant messaging, chat conferencing, etc. The infrastructureless nature of mobile ad hoc networks (called MANETs for short) rules out the possibility of deploying services based on dedicated central entities. And even if a node volunteers to play the role of a central server, the global service provided by this node will not scale and will soon suffer from bad performances due to interruptions caused by the mobility of nodes, network splits and bandwidth scarcity. Furthermore, MANET nodes are end users having, usually, modest resources. Hence, a single node cannot handle the global load of the service. A decentralized approach like peer-to-peer is a good candidate solution to be adopted in such environments. Users of a peer-to-peer service, called peers, organize themselves in an overlay network by connecting to each other via logical links across the other nodes of a MANET. For example, a P2P content sharing application like BitTorrent [1]distributes the data-transfer load among all the peers interested in the same content. Peers who receive some content pieces are responsible of disseminating them to the rest of the P2P network.

The problem is that although P2P applications are designed to be completely decentralized, most of them rely on central servers in some of their functionalities. They generally use servers for discovering and updating the information on the members of their overlays. In fact, a P2P application needs to be permanently informed about the set of peers interested in the same service in order to adjust its overlay by accounting for the arrival and departure of peers. For instance, in BitTorrent, each peer is asked to contact periodically a central rendezvous server called Tracker to get up-to-date information about the members of the Torrent. This way the peer can choose the other peers with whom to exchange pieces of the content. Globally, actual architectures like BitTorrent and Instant messaging mainly focus on distributing the data plane but keep centralized the membership management plane. This way they keep the control on the service they provide while fully profiting from the advantages of the P2P semantic in distributing the load of the data plane.

The presence of this centralized component in some Internet P2P applications makes it difficult to run them in MANETs. The use of a central node as a membership information directory raises the concern of the single point of failure of the service. In fact, unlike an Internet server, a MANET node has limited resources and cannot handle frequent solicitations. Subsequently, it becomes a bottleneck for the service and overwhelms the underlying network which is known for its scarce and shared resources. Moreover, mobility of nodes, shadowing, churn and network splits can be major factors of interruption of the communication with the central server node. One can imagine the scenario where the network is partitioned into two completely disconnected parts. This means an interruption of the service in the part that does not contain the membership server.

In this work, we study the membership management issue in MANET and propose a standalone membership management protocol that answers the needs of a variety of P2P services when they are deployed in these networks.

It allows P2P applications designed for the Internet to migrate to MANET without any significant changes in their service overlays. They can use our protocol to construct and share common knowledge about their overlays equivalent to what is provided by central servers in the Internet. Our protocol overcomes the limitations of the central-server solution and takes into account the constraints of mobile ad hoc networks. It relies on a peer-to-peer approach and is then completely decentralized. According to our protocol, peers organize themselves in a shared tree dedicated for disseminating membership information. Events like new arrivals or departures of nodes are announced on the tree so that each node can keep permanently an up-to-date list of the members of the P2P service and of related information. Using ad hoc routing information, they construct and adapt their logical links in the membership tree with respect to the current topology of the network. Their goal is to minimize the length of the tree(in terms of number of wireless hops) so as to reduce the membership traffic and the overhead on the underlying network. The membership tree can be seen as a distributed minimum spanning tree connecting peers of the service. We propose fully decentralized mechanisms that allow peers to adapt the membership tree structure to the frequent changes of the underlying network caused by nodes' mobility. Moreover, our protocol addresses the partitioning issue in MANET. We achieve this by the help of a new and simple mechanism that allows peers to benefit from the information provided by the underlying routing protocol for discovering peers which come from separate overlays. This allows two or more separate trees for membership management belonging to the same service (e.g. same content in BitTorrent) to efficiently merge together forming a new covering information tree.

In the literature, many works have been conducted to implement peer-to-peer applications in MANETs [11] [12] [13]. We refer to the related work section for a brief description of these implementations. The majority of them do not study independently the membership management issue. In fact, the cost of the membership management is often ignored compared to the cost of the data traffic. More importantly, these works do not provide a solution for the network splits in MANETs that are caused by nodes' mobility and the finite range of the wireless. If not handled correctly, these splits may lead to an interruption of the service for some peers and an underutilization of the power of the peer-to-peer paradigm in MANETs.

To validate the efficiency of our protocol compared to classical solutions, we add a module to the NS-2 network simulator [2] and conduct extensive simulations. The performance of a membership management solution can be measured in terms of the volume of traffic it generates and the level of freshness of the knowledge about the members of the P2P service it allows. As there is a tradeoff between increasing the freshness of membership information and diminishing the cost of the management, we define appropriate metrics to measure the efficiency of the compared solutions in both regards. The comparison of our protocol to client-server, flooding and multicast-based solutions shows that it achieves lower network overhead while ensuring a better membership information freshness.

The remainder of this paper is organized as follows. Section II overviews the related work. Section IV explains the design of our protocol and includes a detailed presentation of its algorithms. Section V is a performance evaluation of our protocol compared to other solutions. Section VI summarizes the paper and gives some ideas on our future work.

## II. Related work

In this section, we overview the body of the literature relevant to our membership management problem. First, we describe the efforts done to manage the membership of P2P systems in the Internet. Then, we present some P2P overlays implemented in MANET. Finally, we study P2P multicast overlays in MANET for the purpose of underlining the similarities and the differences that exist between membership management and multicast.

### *A. Membership management in the Internet*

Many membership management techniques have been proposed for the Internet. They can be subdivided into two categories: those decoupling the P2P data plane from membership management and those coupling them together. One can mention the client-server architecture used by BitTorrent to track peers as a solution that decouples the two functionalities. In BitTorrent, each peer contacts periodically a central rendezvous server named Tracker in order to update its list of peers. In parallel and to distribute the server functionalities, mechanisms based on Dynamic Hash Tables (DHTs) have been also introduced to provide peer-to-peer applications with membership information without relying on one single server. For example, P2PSIP [14] organizes nodes into a structured DHT-based overlay and allows ordinary peers having abundant capacities to become servers. Ordinary peers locate servers by DHT lookup functions. DHT-based solutions are efficient in the Internet since the graph of communication is totally meshed and the bandwidth is abundant. In a MANET these properties unfortunately do not hold. The network may split into separate clusters and nodes serving as DHT servers remain the bottleneck. Other P2P protocols do not take into consideration any quality of service criteria when constructing their overlays and so they use the same structure to do both peer discovery and data dissemination. Content-based routing P2P networks [5] are examples of these techniques. In general, when quality of service is a concern, it is better to decouple the membership management from the data overlay construction. Some other works [15] address the scalability problem inherent to membership management by deploying gossiping techniques. The solution proposed is to contact a random sub-set of the peers and to exchange with them known information on other peers. This technique generates random graphs over which peers exchange their knowledge about the service overlay. It is an acceptable option when the knowledge of a sub-set of peers is sufficient for a good P2P service and when the communication between faraway nodes is not constrained by physical connectivity. Otherwise, the overhead on the underlying network will be very important and the P2P application will suffer from bad performances.

### *B. P2P overlays in MANET*

In general, one can divide the design space of P2P overlays in MANET into four subspaces: Non-structured and layered design [11], non-structured and cross-layer design [12], structured and layered design [13] and structured and cross-layer design [13]. Cross-layer design approaches have been introduced because nodes are both end-users and routers. They suppose that P2P applications operate both at the network layer and at the application layer. Structured approaches suppose that peers are organized following a structured virtual topology. For example, the structure can be a DHT allowing service lookup. Generally, the peers responsible of providing the service can be discovered through a routing in the DHT network. Unfortunately, these DHTs are difficult to adapt to the underlying topology.

### *C. P2P multicast overlays in MANET*

The problem of constructing a P2P multicast protocol for MANET has some common challenges with the problem of constructing a protocol for P2P membership management as in our case. Indeed, multicast protocols need in their functioning a membership management component to track the MANET nodes interested in the session. The problem is that multicast protocols often aim at optimizing the data transfer plane and neglect the signaling plane. The energy spent on signaling is compensated by the efficiency of the data plane itself. In our case, we only focus on the dissemination of the membership information itself, which could be seen as only having the control packets of a P2P multicast overlay without the data. The latter is clearly suboptimal since multicast protocols do not seek the optimization of the flow of control packets. Moreover, some existing multicast protocols are centralized [9] or require global knowledge which we want to avoid [8]. We add that there is no multicast-based solution for the problem of network splitting.

To optimize the data transfer plane, multicast protocols proposed for MANET were constructed following two approaches: Protocols based on meshed overlays and protocols based on tree overlays. Meshed overlays are non-structured networks. They represent random graphs linking nodes of the network. This kind of overlays offers more connectivity and more robustness by maintaining redundant paths between nodes. Nevertheless, the meshed topology is not efficient in MANET due to the overhead caused by duplicated transmissions of packets on redundant paths. Unlike meshed overlays, the tree topologies are very efficient in the MANET environment as they result in low load on the network by avoiding path redundancies. But they are less robust and require specific mechanisms to adapt to the frequent changes in network topology. Our protocol adapts a minimal-cost tree structure while making it adaptive and resilient to network splits.

Here are some examples of overlay multicast protocols recently proposed for MANET.

- **PAST-DM** stands for Progressively Adapted Sub-Tree in Dynamic Mesh [8]. Peers in PAST-DM first organize themselves in a mesh network and then each of them, knowing the topology of this mesh, computes in a centralized way a minimum spanning tree. Each peer discovers its neighbors in the meshed graph by broadcasting messages in a limited scope. This discovery is done periodically in order to adapt the mesh to the underlying topology. Neighbors in the mesh are linked through unicast tunnels in order to exchange link-state information allowing the computation of the spanning tree. When a peer leaves the overlay, the information on its departure propagates via the unicast tunnels until it reaches all the members of the overlay. The periodic exchange of link-state information is very costly and the computation method is not optimal since it must be done periodically by each node.
- **TrAM** [9] is a core-based multicast protocol. Each peer that wants to join the overlay must execute a parent discovery phase by flooding a QueryParent message. The peers that receive the QueryParent message answers by sending ParentAdvertise packets. After sometime, the new peer collects many answers from peers already in the overlay. It then chooses the nearest peer as its parent node. Periodically, peers flood QueryParent messages in order to discover the best possible parents and as a consequence the quality of the tree is ameliorated. When a peer wants to leave the overlay, it informs its current parents in the tree. These parents trigger parent discovery procedures in order to discover new parents. This multicast protocol results in important parent discovery traffic since it is done by all peers and in a periodic manner.
- **MOST** (Multicast Overlay Spanning Tree Protocol) [10] is an overlay multicast protocol based on the construction of a minimum spanning tree. This protocol requires imperatively the use of the OLSR routing protocol. In fact, each peer uses the topology information provided by OLSR in order to compute an optimal spanning tree. This tree is recomputed periodically to adapt to the changes in the topology and in the members of the overlay. Peers flood periodically JOIN messages including the addresses of the multicast groups they belong to. Each peer maintains a peer list per multicast group. If it does not receive any JOIN message from one of the peers during a specific period of time, it deletes it from the lists of peers of its multicast groups. Here also the cost of flooding JOIN messages periodically is very important and the solution is routing protocol dependent.

## III. Graph theory concepts

In this section, we present some graph theory concepts that have been useful in the design of our membership management protocol. Let $G(V, E)$ be a graph. $V$ and $E$ are respectively the set of vertices and the set of edges of the graph. One calls:

- **Cycle:** A cycle is a subset of edges that forms a path such that the first node of the path corresponds to the last one.
- **Cut:** A cut is a partition of the vertices of the graph into two disjoint sets $S$ and $T$. Any edge $e(u; v) \in E$ with $u \in S$ and $v \in T$ is a cut edge.
- **Tree:** A tree is a graph in which any two vertices are connected by exactly one path. Alternatively, any connected graph with no cycles is a tree.
- **Spanning tree:** Given a connected, undirected graph, a spanning tree of that graph is a subgraph which is a tree and which connects all the vertices together. A single graph can have many different spanning trees.
- **Minimum spanning tree:** One can assign a weight to each edge of a graph. The weight of a spanning tree can be then computed as the sum of the weights of the edges in that spanning tree. A minimum spanning tree or minimum weight spanning tree is then a spanning tree whose weight is less or equal than the weight of every other spanning tree. More generally, any undirected graph (not necessarily connected) has a minimum spanning forest, which is a union of minimum spanning trees for its connected components.

The following properties of a minimum spanning tree have been profitable for the development of our protocol:

- **Cycle property:** For any cycle C in the graph, if the weight of an edge e of C is larger than the weights of other edges of C, then this edge cannot belong to a minimum spanning tree.
- **Cut property:** For any cut C in the graph, if the weight of an edge e of C is smaller than the weights of other edges of C, then this edge belongs to all minimum spanning trees of the graph.

## IV. Specification of the membership management protocol

In this section, we describe our protocol for P2P membership management in MANET. Our protocol constructs a spanning tree to be used for the exchange of membership information among peers in the P2P network. We want this tree to match the topology of the underlying network in order to minimize the cost of the dissemination of membership information among peers and to ensure the freshness of the lists of members maintained by each peer. As optimality is needed, we propose to construct a minimum spanning tree in terms of number of hops, covering all peers of the underlying routing graph (some MANET nodes might not be P2P members). This guarantees a minimum cost of the membership information dissemination in terms of number of hops, transmissions and power. We design efficient and distributed mechanisms to track the intermittent connections and disconnections of the MANET nodes. Moreover, because MANET nodes are continuously moving and so tree weights are subject to changes, our protocol needs to restructure the tree when needed in order to maintain its optimality property. Centralized algorithms, as the well known Kruskal algorithm [3], are not good in our case because they require global knowledge and that each peer calculates its own spanning tree. Since the minimum spanning tree is not unique, peers might then calculate different spanning trees which disconnects the service overlay. Our protocol is based on a completely distributed approach which guarantees the uniqueness of the tree by making all decisions locally. Optimality is ensured by satisfying the cycle and cut proprieties in Section III.
Another important problem that we consider in the construction of our adaptive tree is the fact that MANET is frequently subject to network partitioning. So we add to our protocol a specific technique to merge separate trees belonging to the same P2P network together when communication opportunities occur. The following paragraphs describe our protocol and the ideas it implements to construct and update the membership spanning tree.

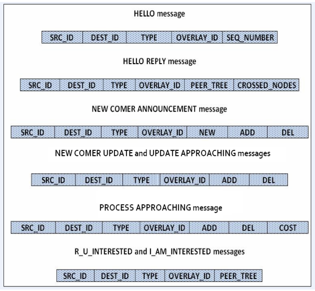

Fig. 1. Packet format

TABLE I
Packet fields description

| Field name | Field description |
|---|---|
| SRC_ID | The identifier of the peer sending the message |
| DEST_ID | The identifier of the peer that will handle the message |
| TYPE | The type of the message (e.g. HELLO) |
| OVERLAY_ID | The identifier of the service overlay |
| SEQ_NUMBER | This field indicates the sequence number of the message. |
| PEER_TREE | A string representing the tree of the node sending the message. |
| CROSSED_NODES | A set of nodes tagged for the network partitioning awareness. |
| NEW | The identifier of the node joining the membership overlay. |
| ADD | A list of logical links to be added to the current tree. |
| DEL | A list of logical links to be removed from the current tree. |
| COST | The weight of the most costly logical link in a cycle. |

### A. Joining the membership tree

We suppose that each P2P service has a unique identifier (for example the file ID in BitTorrent). Knowing this identifier, a node that becomes interested in the service initiates a join procedure. This procedure can be divided into two phases: discovering the nearest peer and disseminating the new arrival information to all other peers. This can trigger the restructuration of the membership tree to maintain its optimality.

*1) Discovering the nearest peer:* In order to discover a first attachment point to the membership tree, we propose to use a simple flooding technique with controlled scope. The new member floods a `HELLO` message in its one hop neighborhood (TTL=1) and then waits for HELLO REPLY messages, sent in unicast, from any member of the P2P network located at one hop. In case there is no answer, the new member increments exponentially the value of the TTL of the HELLO message and waits again for at least one HELLO REPLY. If the maximum TTL is reached and no answer is received, the node considers that the service is not provided in the network and that it is up to it to construct a new membership tree. If it does, it will automatically become the only node in the new spanning tree. Now, if an answer is received, the service already exists and the new member gets in the HELLO REPLY message a copy of the current tree with the list of members and other useful information as the canonical name and description. We underline that in our method a peer does not need to know the cost of the edges of the tree, it only knows which peer is connected to which other peer. The format of the HELLO and HELLO REPLY messages is depicted in Table I and in Figure 1.

Using its current routing table, the new arriving peer compares the costs in number of hops to other peers in the tree. This comparison allows it to identify the closest peer to it. If this closest peer is different than the current one, the new peer should then change its connection in the spanning tree to attach to this closest peer as required by the cut property described in Section III. This property requires that the connection to add is the one having the lowest cost in the cut formed on one side by the old tree and on the other side by the new arriving peer. In practice, after identifying this closest peer, the new arriving peer sends in unicast a simple CONNECT ME message to it. Receiving this message, the nearest peer triggers a new arrival information dissemination phase on the old tree. This phase is coupled with an adaptation of the tree, to be described next, in order to conserve its optimality.

*2) New arrival information dissemination and tree adaptation:* When a member of the tree receives a CONNECT ME message from a new joining peer, it adds this peer as a child node. This modification of the tree is then disseminated to the other peers to trigger any necessary modification that keeps the tree optimal. The new parent sends to all its neighbors in the tree (its parents and its children), except the new arriving peer, a NEW COMER ANNOUNCEMENT message containing the identifier of the new peer and the modifications it has made on the tree. We refer you to Figure 1 and Table I for details on this message. Every peer that receives the NEW COMER ANNOUNCEMENT message updates its knowledge about the tree and verifies whether it can modify some of its logical links to improve its connectivity to the tree next to this new arrival. This modification is described in the following. For now and after making these local decisions, it informs its neighbors in the tree, except the peer which has sent the NEW COMER ANNOUNCEMENT message. It does that by sending a new version of the NEW COMER ANNOUNCEMENT message adding, eventually, its own changes. Upstream peers that have already seen the NEW COMER ANNOUNCEMENT message are informed by the modifications through a NEW COMER UPDATE message, which contains the logical links that the peer has added or removed from the tree. The NEW COMER UPDATE message differs from the NEW COMER ANNOUNCEMENT message by the fact that it does not trigger any modification of the tree. A peer receiving this former message just updates its knowledge about the tree. In this way, all peers are aware of the new arrival and the tree is restructured in parallel.

The decision that a peer must make when it receives a NEW COMER ANNOUNCEMENT message is based on a simple verification of the cycle property described in SectionIII. The cycle to consider is the cycle formed by the logical links on the path of the current tree starting from the intermediate peer making the decision to the newly joining peer, and by adding the direct logical link between both peers. If any optimization is possible, it will result in cutting the logical link through which the peer received the NEW COMER ANNOUNCEMENT message and adding the logical link to the newly joining peer. This way the cost of the tree is always kept minimal. One can notice that all the decisions are made locally and in a distributed manner without compromising global optimality.

### *B. Adapting the membership tree to mobility of nodes*

Due to the mobility of MANET nodes, the distances between the peers of the membership tree vary in time. If the spanning tree is not adapted to these movements, it will quickly lose its optimality property. One can distinguish four possible movements of peers:

- Two peers that are neighbors in the tree can get closer. In this case, the weight of the spanning tree becomes smaller but it remains a minimum spanning tree. This movement has no impact on the structure of the tree.
- Two peers that are not neighbors in the spanning tree get farther from each other. In this case, the weight of the tree does not change and there is no decision to be taken.
- Two peers that are neighbors in the spanning tree get farther from each other. The weight of the current tree increases which means there might exist a better tree to be identified.
- Two peers that are not neighbors in the spanning tree get closer to each other. In this case, the cost of the current spanning tree does not change but there might be another spanning tree with a smaller weight. This movement requires, eventually, an adaptation of the spanning tree seeking for the existence of an optimal one.

In the following paragraphs, we describe how we adapt the membership tree in response to the two latter movements impacting its optimality:

*1) Two neighbors in the tree get farther:* If two peers that are neighbors in the spanning tree get farther due to mobility, this leads to two possible situations. The first situation is that one of these peers or maybe both will get closer to other peers of the tree. Here, the tree adaptation can be done by applying the approaching adaptation procedure which we describe in IV-B2. The second situation is that no one of these two peers gets nearer to other peers, in this case no better spanning tree can be found and no adaptation of the tree is needed. Hence, the problem raised by neighbors getting farther from each other can be transformed into a simple approaching problem and solved by the solution we come up for the latter one.

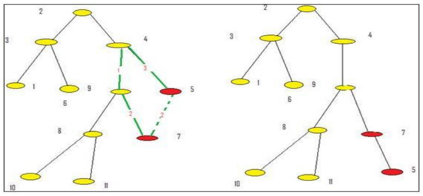

Fig. 2. Adapting the tree after peers 5 and 7 get closer due to mobility

*2) Two peers that are not neighbors in the tree get closer:* Let $P1$ and $P2$ be two peers that are not neighbors in the spanning tree. Suppose that the cost of the physical direct path between these two peers becomes smaller due to the mobility of nodes. Take the cycle formed by the actual logical path (in the tree) from $P1$ to $P2$, and by adding the logical link that connects directly $P1$ to $P2$. If there is a logical link $L$ in this cycle such that $cost(L) > cost(P1, P2)$, the cycle property described in Section III indicates that the logical link $(P1, P2)$ must belong to the minimum spanning tree. So the actual tree should be adapted in such a way to replace the logical link $L$ by the logical link $(P1, P2)$. We illustrate this in Figure 2, where peers 5 and 7 represent respectively peers $P1$ and $P2$. The figure plots the minimum spanning tree before and after the adaptation. This adaption procedure must be executed in a distributed and triggered manner. Each peer in the tree tracks continuously other peers and decides if another peer, which is not its neighbor in the current tree, becomes closer to it than normal. Here, it forms the cycle between it and this peer and initiates a procedure of identification of the most costly link in the cycle. Between the two peers, the one having the lowest identifier initiates the procedure. This is done by circulating a PROCESS APPROACHING message on all logical links of the cycle. Each peer in the cycle adds its logical link to the next peer in the cycle as being the link to be removed (DEL field in the message) if this link is more costly than the link it finds in the message. It also updates the field COST in the message because peers only know the costs of their own logical links. This procedure is repeated until the message returns to its original sender which then can decide which link to be removed and which link to be added. This modification is then disseminated to all peers. The peer sends an UPDATE APPROACHING message to its neighbors in the tree which forward it to their neighbors and so on. The messages are described in Figure 1 and Table I.

### *C. Leaving the membership tree*

Adapting the membership tree after the departure of peers is very important for the efficiency and the uniqueness of the tree and for the freshness of the membership information. We ask every peer that decides to leave the P2P service to inform its logical neighbors in the spanning tree by sending an I AM LEAVING message to them. Excepting for leafs, this departure will result in the decomposition of the tree into two separated sub-trees. The first sub-tree represents the children of the leaving peer and the second one its parents. To reconnect the tree, the child of the leaving peer having the highest identifier connects to its parent and becomes the parent for the remaining children. This way, a new spanning tree is formed. The problem is that this new tree may not be optimal. The optimal is reached by having the peers apply the normal approaching adaptation procedure described earlier in paragraph IV-B2. Note how this procedure is important for our solution to always rewire the tree in a way to ensure its optimality. All modifications made on the old tree after the departure, are disseminated to all peers of the tree together with the identifier of the departing peer. Figure 3 gives an example of adjustment of the spanning tree after the departure of peer 5. Sometimes a peer can leave the service overlay improperly; in this case, it is the duty of the first neighbor detecting this phenomenon to trigger the tree adaption procedure.

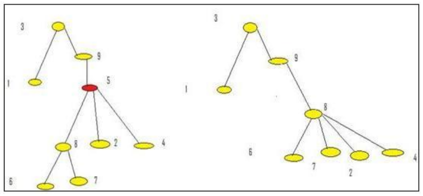

Fig. 3. The membership tree before and after peer 5 leaves

### *D. Network split awareness*

Due to the high dynamicity and mobility of MANETs, the network can split into different disconnected clusters. These clusters can merge again into one or more larger clusters. So one can imagine the scenario where one P2P network is split into two or more networks because of the network splitting. Another scenario is that two or more membership trees constructed separately in different clusters but belonging to the same P2P network. At the first merging opportunity, it is very important to connect together the different partitions of a membership tree in one large and efficient tree. That is why; we add to our membership protocol a simple mechanism allowing peers to discover other peers of the same service coming from other partitions when they get close to each other. After connecting to the current minimum spanning tree, each peer observes its routing table and tags network nodes that are not interested in the same service. Then, using its routing table, it tracks continuously the appearance of non tagged nodes in its neighborhood. We choose the neighborhood of a node in the P2P network to be equal to the maximum number of hops to one of its direct neighbors on the spanning tree. A new node not tagged and not belonging to the same membership tree is a good candidate to be asked whether it belongs to the same service and comes from another cluster. The peer sends to this newly detected node a message R U INTERESTED in order to ask it whether it is interested in the service. If it receives no answer from that node then it tags it as a not interested node. The tagging information is disseminated to all the peers in order to reduce the number of R U INTERESTED messages. When a node receives an R U INTERESTED message and if it is interested in the service, it answers the source by sending an I AM INTERESTED message. In this case, the two trees maintained by the two peers need to be merged together. To ensure efficient merging, the peers of the smallest tree apply the join procedure in order to connect to the biggest tree.

## V. Performance evaluation

In this section, we study the performance of our protocol in comparison to classical membership management approaches. The validation is based on extensive simulations run with our implementation of the different membership management methods in the NS-2 network simulator [2].

### A. Performance metrics

Let $P_t$ be the set of peers interested in the service at an instant $t$ and let $p_t$ be the cardinal of this set of peers. When a peer is running a membership management approach, it maintains at an instant $t$ a set of peers $N_t$, consisting its view of the members of the P2P service. Let $n_t$ be the cardinal of this set. Among these $n_t$ peers, there are $t_t$ peers belonging to $P_t$ and $f_t$ not belonging to it (e.g. due to peers which have left). During a specific measurement time (namely the simulation time for us here), peers exchange messages between them in order to discover the interested peers and update their knowledge about them. Let $C$ be the cost in number of hops over paths crossed by the exchanged messages during a fixed period of time. The importance of this cost depends on the used membership management. However, this cost does not take into consideration the freshness of information maintained by the peers and then it is not enough to decide whether a method is appropriate or not. In fact, one can spend a very low cost and have a lot of pollution in its knowledge about the peers. That is why we propose another metric named cost corrected by freshness of information which we note $C_f$. This cost is also a global metric computed during a fixed time. After each $T_s$ seconds, one takes a snapshot of the P2P network and measures the value of $p_t$ and computes $\overset{\wedge}{n}_t$, $\overset{\wedge}{t}_t$ and $\overset{\wedge}{f}_t$, the average values of $n_t$, $t_t$ and $f_t$ over peers interested in the P2P application. At the end of the simulation time, one computes $\tilde{p}$, $\tilde{n}$, $\tilde{t}$ and $\tilde{f}$ the average values of $p_t$, $\overset{\wedge}{n}_t$, $\overset{\wedge}{t}_t$ and $\overset{\wedge}{f}_t$ over all measured samples. The cost corrected by freshness of information $C_f$ is the ordinary cost to which we add two terms. The first term accounts for the cost that the P2P network should pay to discover the $\tilde{p} - \tilde{t}$ missing members. This term can be easily calculated considering that the members of the P2P application have paid $\frac{C}{\tilde{n}}$ to discover a peer. Hence, the term of lack of information cost will be equal to $\frac{C(\tilde{p}-\tilde{t})}{\tilde{n}}$. The second term to be added to the ordinary cost is a term accounting for the pollution existing in the knowledge of the peers. We consider that one pays the same cost to discover an interested peer or to remove an idle one. That is why we take this term equal to $\frac{C\tilde{f}}{\tilde{n}}$. The following formula computes the cost metric corrected by the freshness of information:

$$C_f = C\left(1 + \frac{\tilde{p} - \tilde{t}}{\tilde{n}} + \frac{\tilde{f}}{\tilde{n}}\right) \tag{1}$$

### B. Scenario description

To conduct our simulations, we consider a MANET composed of 50 nodes moving inside a bounded area of width 100m and length 500m following the Random Waypoint mobility model. The speed and the pause time of each node are taken equal to 2m/s and 30s respectively. The nodes connect to each other using the 802.11 MAC layer with the RTS/CTS-Data/ACK mechanism enabled. The range of transmission is fixed to 50m and the data rate is set to 1 Mb/s. For ad hoc routing, we use the proactive OLSR protocol [4]. To simulate a dynamic membership, we suppose that a node has two states: The first state is an idle state where it is not interested in the P2P service. The second state is the one where it becomes interested in the P2P service. The membership of a node follows then an ON/OFF process until the end of the simulation. The durations of the ON and the OFF states follow an exponential distribution of parameter $\lambda$ and $\mu$ respectively. We define the density of the P2P overlay as the number of nodes interested in the P2P network divided by the total number of MANET nodes, One can easily show that this density is on average equal to $\frac{\frac{1}{\lambda}}{\frac{1}{\lambda}+\frac{1}{\mu}}$. When not stated, the density is taken equal to 50% by assuming that both $\lambda$ and $\mu$ are equal to 500s. The simulation duration is set to 3600s. The sampling period $T_s$ used to compute the corrected cost is chosen equal to 10s.

### C. Comparative study

We compare, through extensive simulations, our protocol to four classical methods for membership management: a client/server method, a flooding-based method, a multicast-based method and a non-adaptive tree method.

- **Client/server method:** The classical client/server method supposes that each peer contacts periodically a server to update its knowledge about the members of the P2P application. In our simulations, a random node plays the role of the server. Figure 4 plots the real cost in number of hops-messages as a function of the period at which the peers contact the server. It shows that this cost is proportional to the inverse of the contact period. The same figure also plots the cost corrected by the freshness of the membership information. One can easily notice that this corrected cost is higher than the ordinary cost. Contrary to the real cost, it does not continuously diminish when the period of contacting the server increases. In fact, the freshness of information decreases with the increase in the contact period, which is accounted for in our corrected cost metric. In the following simulations, the contact period is set to 400s which according to the figure yields the best performances.

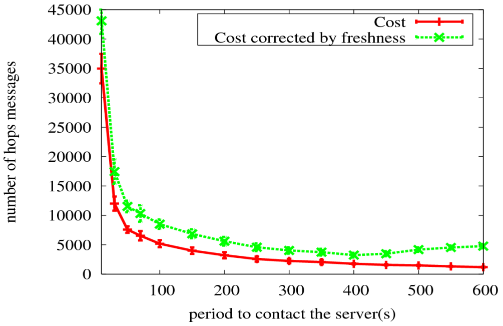

Fig. 4. Real cost and corrected cost Vs. Period to contact the server

- **Flooding-based method:** Peers advertise their arrivals and their departures to other interested peers by physically flooding the network. Two types of sequenced messages are used for this purpose: peer-joining and peer-leaving messages. Receiving such a message, each node in the MANET forwards it to its physical neighbors if it is seen for the first time, otherwise it is discarded. During the broadcast, if a MANET node is interested in the P2P service, it updates its membership information. The cost of the membership management is equal to the number of hops crossed by the flooded messages.
- **Multicast-based method (PAST-DM):** To compare with multicast, we use the PAST-DM protocol known for its efficiency in MANET. We refer you to the related work section for a detailed description of this protocol and of its membership management mechanism. In our simulations, we implement for PAST-DM the exchange of link state messages, the JOIN/LEAVE messages and the messages to discover peers in the near neighborhood. The period to exchange the link state tables is set to 30s in order to match the value of the pause time for nodes' mobility.
- **Non-adaptive tree method:** This method is very similar to our membership management protocol. It implements the same algorithms but it does not adapt the constructed spanning tree to the topology and dynamicity of the network. In fact, a joining peer does not connect itself to the nearest peer but to the first responding peer and the constructed tree is not adapted to the topological changes of the underlying network. Hence, the constructed tree is a sub-optimal spanning tree. Our aim is to prove the need for topology awareness.

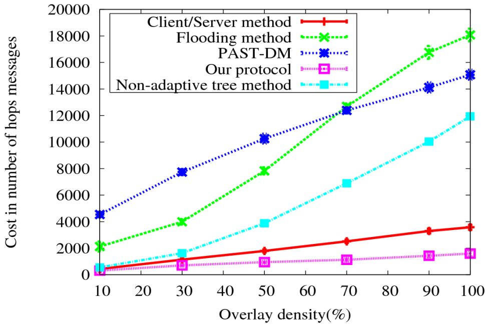

Fig. 5. Real cost Vs. Overlay density

We begin our comparison by analyzing the impact of the overlay density on the real cost of the membership management. The results for the four methods are presented in Figure 5. The figure shows that the cost of the flooding-based method increases linearly with the number of interested nodes in the network and is quite high. This is due to the fact that information about arrival and departure of peers is flooded in the entire network, creating a large number of redundant messages. In contrast, we observe that the cost of our protocol increases slowly with the overlay density while staying very low. One can explain this behavior by the fact that the expanded-ring technique used by our protocol for discovering peers guarantees a low cost in dense overlays. Moreover, update messages circulate along shortest paths of the minimum spanning tree without generating redundant messages. Although PAST-DM implements a controlled flooding technique to overcome the limitations of classical flooding, periodic updates increase dramatically its membership management cost as the overlay density increases. Finally, unlike our protocol, the non-adaptive tree method does not scale when the P2P network grows because of the sub-optimality of the weights of the tree branches.

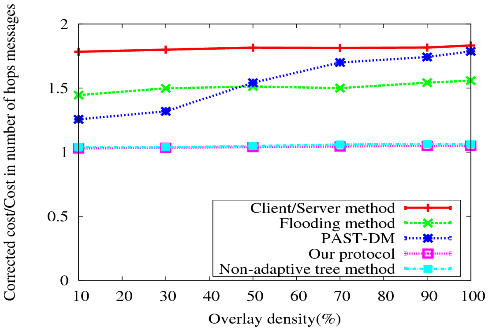

Fig. 6. Ratio of corrected cost to cost Vs. Overlay density

As a second step, we examine the freshness of membership information of the different methods. We plot in Figure 6 the ratio of the corrected cost to the ordinary cost as a function of the overlay density. The higher this ratio is, the more out of date the peers are. A ratio equals to 1 means that peers have correct knowledge of the P2P service members over time. One can notice that our protocol and the non-adaptive tree method achieve a ratio value very close to 1. In fact, in these two methods, triggered updates and the tagging technique allow an efficient and immediate information dissemination among peers. As expected, the client/server mechanism achieves a quite high ratio, even in sparse overlays. This confirms the idea that the client/server method does not scale in wireless environments. Concerning PAST-DM, update information is gradually propagated in the network through iterative exchanges between peers. Hence in dense overlays, where neighbors are physically close to each other, information needs several periods to reach all the peers. This explains the high cost ratio and the increasing trend seen in Figure 6.

In Figure 7, we plot the cost corrected by freshness of the membership information as a function of the overlay density.

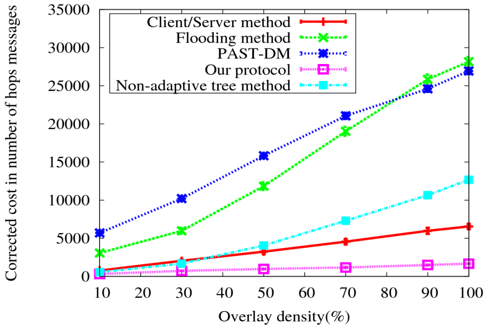

Fig. 7. Corrected cost Vs. Overlay density

By comparing the different methods in regard of this metric, one can decide which one is better than the others in terms of both network overhead and freshness of information. The figure shows that our protocol outperforms the other methods as it achieves the lowest network overhead while keeping a very high level of freshness of the membership information. Unlike our protocol, the non-adaptive tree method has good freshness of information but pays a much higher cost for the overlay construction as we have seen in previous Figures 5 and 6. The flooding-based method and PAST-DM achieve higher corrected costs as they have both higher network overhead and bad freshness of information.

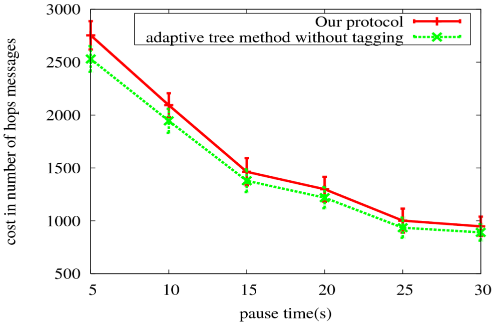

Fig. 8. Split awareness: Cost Vs Pause time

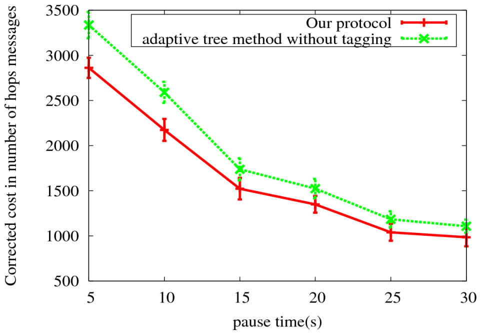

Fig. 9. Split awareness: Corrected cost Vs Pause time

The last set of simulations aims to study the efficiency of our solution for overlay splits versus the frequency of topology changes by varying the pause time of nodes from 5 to 30s. A low value of pause time means frequent topology changes and more probable network splits. We evaluate the capacity of our protocol in handling network splits by simulating it in two modes: a mode that enables the splits' awareness mechanism and a second mode that disables it. Figure 8 shows that the extra cost for handling network splits is relatively small even for low values of the pause time. However, Figure 9 shows that the corrected cost of the splitting-unaware variant of the protocol becomes higher than the corrected cost of the splitting-aware variant thanks to a better freshness of information. Hence, we conclude that our protocol provides an efficient and low-costly solution for MANET partitioning problem.

## VI. Conclusions

P2P membership management is a hard and costly task in MANET. In this work, we propose a scalable, robust and network friendly protocol to construct an adaptive topology-aware tree allowing peers to discover each other and to keep themselves informed about the arrivals and departures of other peers. The proposed protocol is a standalone service that can be used by any application requiring the sharing of up-to-date information among a group of users. Moreover, our protocol minimizes the number of exchanged messages and copes with node mobility and network partitioning, which makes it very useful for applications to know where peers are located in the network and how far they are from each other. The simulations show that our protocol outperforms classical solutions in terms of network load and freshness of information. The future work will be on the integration of this protocol within P2P applications as, for example, the trackerless BitTorrent. Our aim is to study the gain in performance that one would obtain by decoupling the construction of the membership management from the data plane itself.

## References

[1] BitTorrent protocol. http://wiki.theory.org/BitTorrentSpecification.

[2] NS: The Network Simulator, http://www.isi.edu/nsnam/ns/

[3] The Kruskal Algorithm. ttp://en.wikipedia.org/wiki/Kruskal's_algoritm

[4] P.Jacquet, P. Mhlethaler, T Clausen, A. Laouiti, A. Qayyum and L. Viennot Optimized Link State Protocol for Ad Hoc Networks. IEEE INMIC Pakistan 2001.

[5] The Gnutella specification, 2000, http://dss.clip2.com/GnutellaProtocol04.pdf

[6] S.Ratnasamy, P.Francis, M.Handley, R. Karp and S.shenker. A scalable content-addressable networks, ACM SIGCOMM, 2001.

[7] I.Stoica, R.Morris, D.Karger, M.F.Kaashoek and H.Balakrishnan. Chord: a scalable peer-to-peer lookup service for internet applications, ACM SIGCOMM, 2001

[8] Chao Gui, and Prasant Mohapatra, Efficient Overlay Multicast for Mobile Ad Hoc Networks, In Proceedings of IEEE Wireless Communications and Networking Conference (WCNC), March, 2003.

[9] Peter Baumung, TrAM: Cross-Layer Efficient Application-Layer Multicast in Mobile Ad-hoc Networks, Proceedings of IEEE Wireless Communications and Networking Conference, Hong Kong, China, Mar 2007.

[10] G. Rodolakis, A. Meraihi Naimi, A. Laouiti. Multicast Overlay Spanning Tree Protocol for Ad Hoc Networks. In WWIC, Coimbra, Portugal, 2007

[11] L.B. Oliviera, I.G.Siqueira, A.A.Loureiro, Evaluation of ad hoc routing protocols under a peer-to-peer application, WCNC, 2003.

[12] A.Klemm, C.Lindermann, O.Waldhorst. A special-purpose peer-to-peer file sharing system for mobile ad hoc networks, VTC, 2003.

[13] S.M. Das, H.Pucha, Y.C. Hu. Ekta: an efficient peer-to-peer substrate for distributed applications in mobile ad hoc networks. TR-ECE-04-04, Purdue University, 2004.

[14] P2PSIP http://www.p2psip.org/

[15] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah,Gossip Algorithms: Design, Analysis, and Applications, Proc. INFOCOM 2005, March 2005.